Switch Mode Based Deep Fractional Interpolation in Video Coding

Sifeng Xia[1], Wenhan Yang[1], Yueyu Hu[1], Wen-Huang Cheng[2] and Jiaying Liu[1*]
[1]Institute of Computer Science and Technology, Peking University
[2]Department of Electronics Engineering & Institute of Electronics, National Chiao Tung University

***Abstract*—Fractional interpolation is a significant technology in motion compensation of video coding. It generates sub-pixel level reference samples in inter prediction to facilitate temporal redundancy removal between video frames. Recently, some methods explore to introduce the deep learning technique for fractional interpolation and have obtained better compression results. However, existing deep learning based methods still treat fractional interpolation as a traditional interpolation problem but fail to adjust it to the motion compensation scenario. In this paper, we design a switch mode based deep fractional interpolation method to introduce integer pixels of different positions to the interpolation of sub-pixel position samples. By switching between integer pixels of different positions, our method can infer the sub-pixels with smaller variations and achieve better fractional interpolation results. Consequently the motion compensation performance can be further improved. Experimental results have also verified the efficiency of the switch mode based deep fractional interpolation. Compared with High Efficiency Video Coding, our method achieves 2.8% bit saving on average and up to 6.2% bit saving under low-delay P configuration.**

I. Introduction

Motion compensation is an important technology in video coding which utilizes the temporal redundancy among video frames to boost the coding performance. Specifically, during inter-prediction, reference blocks are searched from previously coded frames for a current block which is to be coded. Then, only motion vectors and residues between the reference blocks and the current block need to be coded, which can lead to much bit saving in video coding.

However, due to the discrete spatial sampling of digital videos, signals of adjacent pixels in a video frame are not continuous. Consequently, reference blocks at the integer position may have sub-pixel level motion shifts to the current block. To produce better reference blocks, video coding standards like High Efficiency Video Coding (HEVC) generate sub-pixel level reference blocks from the integer-position reference block with the fractional interpolation technique. The video coding standards commonly adopt fixed interpolation filters for fractional interpolation [1], [2]. This kind of interpolation methods are effective for motion compensation. However, the fixed interpolation filters may fail to fit for natural and artificial video signals with various kinds of structures and content.

Recently, many deep learning based methods have been proposed for image processing problems, *e.g.* image interpolation [3], image denoising [4], [5], and image super-resolution [6], [7]. These works have demonstrated the potential of deep learning technology and generated impressive results. In [3], Yang *et al.* utilized a variational learning network to effectively exploit the structural similarities among images for image interpolation. The deep learning based denoising method [4] utilizes a deep convolutional neural network (CNN) for image denoising by inferring a noise map from the noisy image. Dong *et al.* proposed a super-resolution method called SRCNN [6], which is the first to use CNN for the image super-resolution problem and has obtained significant performance gain over traditional super-resolution methods. In [7], a deeper CNN network is built to achieve a larger receptive field and further improve the super-resolution performance. The residual learning technology is used to facilitate training.

Inspired by the great success brought by the deep learning based methods in image processing problems, some methods explore to introduce deep learning technique into the fractional interpolation problem [8], [9], [10]. Yan *et al.* [8] first proposed a CNN-based interpolation method for the half-pixel interpolation of HEVC. Following that work, Zhang *et al.* [9] used the network architecture of a successful network VDSR [7] in image super-resolution problem to improve the half-pixel interpolation performance in HEVC. These two works only consider the half-pixel interpolation while the quarter-pixel interpolation is not considered. Consequently, a group variational transformation neural network (GVTCNN) was proposed in [10] to improve both the half-pixel and quarter-pixel interpolation performance of HEVC.

Although GVTCNN obtains gain over HEVC and other two deep based methods, it still formulates fractional interpolation as a traditional interpolation problem. In GVTCNN, all the sub-pixels are interpolated by inferring their variations to the top-left integer pixels. However, for some sub-pixels, the motion shift will be smaller if infer them with the integer pixels of other positions. So it is intuitive that the interpolation performance of the sub-pixels will be better if we switch to nearer integer pixels for inference. In this paper, we propose a switch mode based deep fractional interpolation method to infer sub-pixels from integer pixels of various positions. And

*Corresponding author. This work was supported in part by National Natural Science Foundation of China under contract No. 61772043, in part by Beijing Natural Science Foundation under contract No. L182002 and No. 4192025, and in part by the Ministry of Science and Technology of Taiwan under grants MOST-107-2218-E-009-062, MOST-107-2218-E-002-054, MOST-107-2221-E-182-025-MY2, MOST-105-2628-E-009-008-MY3. We also gratefully acknowledge the support of NVIDIA Corporation with the GPU for this research.

978-1-7281-0397-6/19/$31.00 ©2019 IEEE

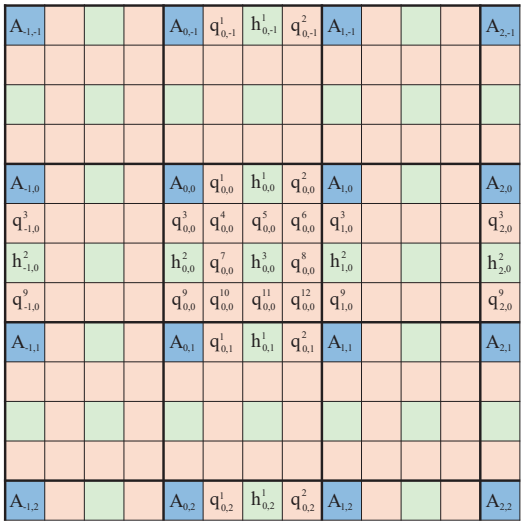

Fig. 1. Positions of different fractional pixels. Blue, green and pink blocks indicate respectively the integer- ($\mathrm{A}_{i,j}$), half- ($\mathrm{h}^1_{i,j}, \mathrm{h}^2_{i,j}, \ldots, \mathrm{h}^3_{i,j}$) and quarter- ($\mathrm{q}^1_{i,j}, \mathrm{q}^2_{i,j}, \ldots, \mathrm{q}^{12}_{i,j}$) pixel positions for luma interpolation.

a coding unit (CU) level rate distortion optimization (RDO) is further designed to choose the best fractional interpolation mode at the encoder side.

The rest of the paper is organized as follows. Sec. II introduces the proposed switch mode based deep fractional interpolation method. Experimental results are shown in Sec. III and concluding remarks are given in Sec. IV.

II. Switch Mode Based Deep Fractional Interpolation

A. Switch Mode based Fractional Interpolation

During the motion compensation in HEVC, blocks from previous coded frames will be used as reference blocks. And fractional interpolation technique generates sub-pixel position samples from the reference blocks to achieve better reference blocks. Fig. 1 illustrates positions of integer pixels and sub-pixels. To be more specific, positions indicated by $\mathrm{A}_{i,j}$ represent integer samples; $\mathrm{h}^k_{i,j}(k \in \{1,2,3\})$ and $\mathrm{q}^k_{i,j}(k \in \{1,2,\cdots,12\})$ denote half-pixel positions and quarter-pixel positions, respectively. Given a reference block I^{A}, whose pixels are regarded as integer samples ($\mathrm{A}_{i,j}$), the half-pixel blocks I^{h^k} and quarter-pixel blocks I^{q^k} are interpolated from I^{A}. With these sub-pixels interpolated, the most similar reference sample is finally selected among integer and sub-pixel position samples to facilitate coding the current block.

In previous work [10], variations between the sub-pixels and the top-left integer pixels are learned to interpolate the sub-pixels, which is the mode 1 as shown in Fig. 2. Specifically, the half-pixel position and quarter-pixel position pixels are formulated as:

$$I^{\mathrm{h}^k_{i,j}} = I^{\mathrm{A}_{i,j}} + \Delta I^{\mathrm{h}^k_{i,j}}, k \in \{1,2,3\}, \quad (1)$$

$$I^{\mathrm{q}^k_{i,j}} = I^{\mathrm{A}_{i,j}} + \Delta I^{\mathrm{q}^k_{i,j}}, k \in \{1,2,\cdots,12\}, \quad (2)$$

where $\Delta I^{\mathrm{h}^k_{i,j}}$ and $\Delta I^{\mathrm{q}^k_{i,j}}$ represent the variations, which are to be learned with CNN from the integer pixels. However, variations of the sub-pixels of different sub-pixel positions

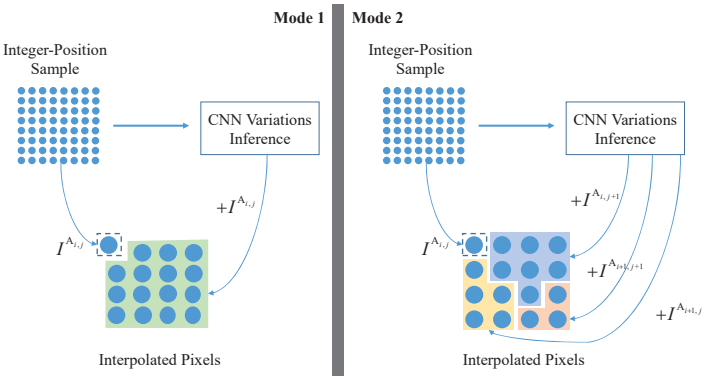

Fig. 2. Illustration of the designed fractional interpolation modes.

differ a lot due to their different distances to the integer pixels. For example, the variation between $I^{\mathrm{q}^{12}_{i,j}}$ and $I^{\mathrm{A}_{i,j}}$ is far larger than the variation between $I^{\mathrm{q}^1_{i,j}}$ and $I^{\mathrm{A}_{i,j}}$. Alternatively, the variation between $I^{\mathrm{q}^{12}_{i,j}}$ and $I^{\mathrm{A}_{i+1,j+1}}$ is usually smaller.

We assume that smaller variations are easier to be learned by the network. So it is significant to introduce integer pixels of more positions to fractional interpolation, by which smaller variations can be found for many sub-pixels. To this end, apart from the existing interpolation mode which infers the variations all from the top-left integer pixels, we further design an interpolation mode that chooses another set of integer pixels for interpolation, which is shown as the mode 2 in Fig. 2. With the designed interpolation modes, the encoder can switch between them to achieve better fractional interpolation performance.

B. Architecture of the Variation Learning Network

To accomplish the designed switch mode based fractional interpolation, a variation learning network is built to infer the variations from the integer position sample I^{A}. The interpolated sub-pixel samples are derived by adding the variations to the corresponding integer position samples according to the interpolation mode. In this subsection, we will introduce details of the variation learning network.

In the network, we use the parametric rectified linear units (PReLU) [11] for nonlinearity between the convolutional layers. Specifically, we define f_k^{out} to be the output of the k-th convolutional layer. f_k^{out} is obtained by:

$$f_k^{out} = P_k\left(W_k * f_{k-1}^{out} + B_k\right), \quad (3)$$

where f_{k-1}^{out} is the output of the previous layer, W_k is the convolutional filter kernel of the k-th layer and B_k is the bias of the k-th layer. f_0^{out} is the input integer-position sample. The function $P_k(\cdot)$ is the PReLU function of the k-th layer:

$$P_k(x) = \begin{cases} x, & x > 0, \\ a_k * x, & x \leq 0. \end{cases} \quad (4)$$

x is the input signal and a_k is the parameter to be learned for the k-th layer. a_k is initially set to 0.25 and all channels of the k-th layer share the same parameter a_k.

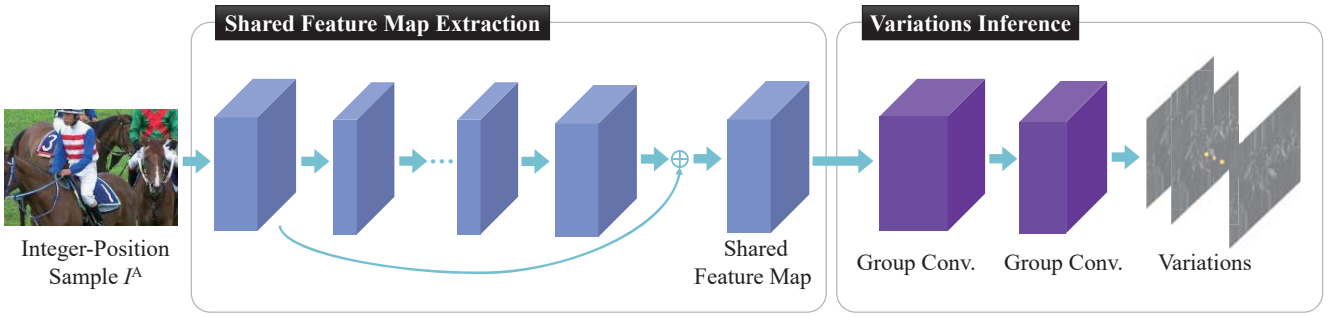

Fig. 3. Framework of the proposed variation learning network. The network first extracts the shared feature map from the integer-position sample. Then the variations that identify the differences between different sub-pixel position samples and integer-position samples are inferred separately from the shared feature map by group convolution. Sub-pixel position samples can be later derived by adding the inferred variations to corresponding integer-position samples.

Fig. 3 shows the architecture of our network. A feature map with 48 channels is firstly generated by convolution from the integer-position sample. And the subsequent are 8 convolution layers with 10 channels which are lightweight and cost less to save the learnt parameters. The 10-th layer later derives a 48 channel feature map. The residual learning technique is utilized here for accelerating the convergency of the network. We add the output of the 1-st layer to the output of the 10-th layer and then activate the sum with PReLU function to derive a 48 channel shared feature map. Later, the shared feature map is copied by concatenation and two group convolution layers are used to infer the variations of different sub-pixel position samples separately from the copied shared feature maps.

With the learned variations, the final interpolated sub-pixel position samples are obtained by adding the variations to the corresponding integer-position samples.

C. Training Details

For fractional interpolation, there exists no ground truth because the sub-pixel position samples in fact do not exist. Consequently, referring to the previous methods, simulated integer position samples and sub-pixel positions samples are sampled from images to form the training pairs. In this paper, we use 400 images in *BSDS500* [12] at size 481×321 and 321×481 to generate the training data. Moreover, due to different sub-pixel levels of half-pixel samples and quarter-pixel samples, training data generation of half-pixel and quarter-pixel position samples is implemented separately with different settings.

Specifically, for half pixel interpolation, 3×3 Gaussian kernels with random standard deviations in the range $[0.4, 0.5]$ are used for blurring to alleviate the artifacts brought by the subsequent sampling process. By dividing the images into 2×2 patches without overlapping, pixels at the top-left of the patches in the raw images are sampled to obtain the integer-position sample. And pixels at other three positions of the patches are separately sampled from the blurred image to derive the sub-pixel position samples.

As for quarter pixel interpolation, the inferred samples are at a smaller sub-pixel level. 3×3 Gaussian kernels with random standard deviations in the range $[0.5, 0.6]$ are utilized. The sampling is performed based on 4×4 patches, where 12 samples are extracted from pixels at $1/4$ or $3/4$ positions vertically or horizontally in the patch.

For the first interpolation mode, the sampled integer position sample is directly used as the input. For the second mode, three shifted integer position samples are additionally derived to provide integer pixels of other positions.

We choose mean square error as the loss function for training. Let $F(\cdot)$ represent the learned network and Θ denote the set of all the learnt parameters. The loss function can be formulated as follows:

$$L(\Theta) = \frac{1}{n} \sum_{i=1}^{n} \| F(x_i, \Theta) + \varphi(x_i, t) - y_i \|^2, \qquad (5)$$

where pairs $\{x_i, y_i\}_{i=1}^{n}$ are the generated ground-truth pairs of integer-position and sub-pixel position samples and n is the total number of the pairs. $\varphi(x_i, t)$ represents the corresponding integer-position sample according to the interpolation mode t.

D. Integration into HEVC

At the encoder side, we can use the target block which is to be coded to select the best reference block. Thus a CU level RDO is implemented here to select the best interpolation method among the traditional interpolation method of HEVC and the designed deep fractional interpolation method.

Specifically, two flags are set based on the rate-distortion costs of different interpolation methods. The first flag is set to decide whether to use the deep based method for fractional interpolation, which is set and coded for all CUs of inter mode. If the first flag is set to choose the deep based method, all the prediction units (PU) in a CU will switch to the corresponding deep fractional interpolation mode according to the second flag.

There are total 4 models trained for interpolating half-pixel and quarter-pixel position samples under two modes. Encoder

TABLE I
BD-RATE REDUCTION OF THE PROPOSED METHOD COMPARED TO HEVC.

Class	Sequence	BD-rate		
		Y	U	V
Class B	Kimono	-1.0%	2.0%	1.6%
	BQTerrace	-6.2%	-3.0%	-3.8%
	BasketballDrive	-3.7%	-0.1%	0.0%
	ParkScene	-1.2%	0.1%	0.2%
	Cactus	-3.0%	-0.3%	-1.0%
	Average	-3.0%	-0.3%	-0.6%
Class C	BasketballDrill	-3.4%	0.0%	0.1%
	BQMall	-3.6%	-1.2%	-1.1%
	PartyScene	-1.7%	-0.6%	-1.0%
	RaceHorsesC	-2.0%	-1.0%	0.0%
	Average	-2.7%	-0.7%	-0.5%
Class D	BasketballPass	-3.7%	-1.7%	-1.2%
	BlowingBubbles	-2.1%	0.6%	0.6%
	BQSquare	-1.2%	1.4%	2.3%
	RaceHorses	-2.8%	-1.3%	-0.7%
	Average	-2.5%	-0.3%	0.3%
Class E	FourPeople	-2.1%	0.5%	0.1%
	Johnny	-3.7%	-0.1%	1.1%
	KristenAndSara	-2.6%	0.9%	0.7%
	Average	-2.8%	0.4%	0.6%
All Sequences	Overall	-2.8%	-0.2%	-0.1%

and decoder will automatically choose the corresponding model according to sub-pixel positions and interpolation modes.

III. EXPERIMENTAL RESULTS

A. Experimental Settings

During the training process, the training images are decomposed into 32×32 sub-images with a stride of 16. The network is trained on the Caffe platform [13]. Adam [14] is chosen as the optimizer for the standard back-propagation. The learning rate is initially set to a fixed value 0.0001. The batch size is set to 128. Models after $100,000$ iterations are used for testing. The network is trained on Titan X GPU.

The proposed method is tested on HEVC reference software HM 16.7 under the Low-Delay P (LDP) configuration. In this paper, only the luma component is interpolated. BD-rate is used to measure the rate-distortion. The quantization parameter (QP) values are set to 22, 27, 32 and 37 for testing. We also compare our method with the single mode based deep fractional interpolation method GVTCNN [10].

B. Experimental Results and Analysis

Table I shows the BD-rate reduction of our method in class B, C, D and E under the LDP configuration. Our method has obtained on average 2.8% BD-rate saving and up to 6.2% BD-rate saving for the test sequence *BQTerrace*. The results demonstrate that the performance of inter prediction is improved with the switch mode based deep fractional interpolation.

TABLE II
BD-RATE REDUCTION COMPARISON BETWEEN GVTCNN AND THE PROPOSED METHOD.

Class	GVTCNN	Proposed
Class B	**-3.0%**	**-3.0%**
Class C	-1.7%	**-2.7%**
Class D	-1.5%	**-2.5%**
Class E	-1.9%	**-2.8%**
All Sequences	-2.1%	**-2.8%**

For the purpose of further verification, we additionally compare our method with the latest single mode deep fractional interpolation method GVTCNN [10], which implements deep fractional interpolation by interpolating sub-pixels all with the top-left integer pixels. For fair comparison, we have re-implemented GVTCNN in HM 16.7. The BD-rate reduction comparison of the Y component bwtween the two methods is shown in table II. Our method is superior to GVTCNN in most classes and obtains 0.7% more BD-rate reduction on average.

TABLE III
RDO RESULTS OF THE TWO DEEP FRACTIONAL INTERPOLATION MODES.

Sequence	Mode 1	Ratio	Mode 2	Ratio
BasketballPass	14705	73.77%	5229	26.23%
BlowingBubbles	15452	76.47%	4755	23.53%
BQSquare	6147	70.70%	2547	29.30%
RaceHorses	22329	85.93%	3657	14.07%

We have also counted the numbers of the CUs that choose the proposed switch mode based deep fractional interpolation method with the sequences in class D. We code the frames of the first 2 seconds of the sequences with QPs 22, 27, 32 and 37. The numbers of the CUs that choose the two interpolation modes are shown in Table III. The choosing ratio of the two interpolation modes is also calculated. It can be seen that a decent amount of CUs choose the mode 2 for deep fractional interpolation.

IV. CONCLUSION

In this paper, we propose a switch mode based deep fractional interpolation method for the motion compensation of video coding. Apart from uniformly inferring the variations to the top-left integer pixel for interpolating sub-pixels, integer pixels of other positions are also introduced in our work to make the variations that are to be learned smaller, which can benefit the fractional interpolation and further improve the motion compensation performance. A CU level RDO is used to help the CU switch between different interpolation methods at the encoder side. Experimental results show that our method has obtained on average a 2.8% BD-rate saving on the test sequences compared with HEVC.

REFERENCES

[1] T. Wiegand, G. J. Sullivan, G. Bjontegaard, and A. Luthra, "Overview of the H. 264/AVC video coding standard," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 13, no. 7, pp. 560–576, 2003.

[2] G. J. Sullivan, J. Ohm, W. J. Han, and T. Wiegand, "Overview of the high efficiency video coding (HEVC) standard," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 22, no. 12, pp. 1649–1668, 2012.

[3] W. Yang, J. Liu, S. Xia, and Z. Guo, "Variation learning guided convolutional network for image interpolation," in *Proc. IEEE Int'l Conf. Image Processing*, 2017.

[4] K. Zhang, W. Zuo, Y. Chen, D. Meng, and L. Zhang, "Beyond a Gaussian denoiser: Residual learning of deep CNN for image denoising," *IEEE Transactions on Image Processing*, vol. 26, no. 7, pp. 3142–3155, July 2017.

[5] K. Zhang, W. Zuo, S. Gu, and L. Zhang, "Learning deep CNN denoiser prior for image restoration," in *Proc. IEEE Int'l Conf. Computer Vision and Pattern Recognition*, 2017.

[6] C. Dong, C. C. Loy, K. He, and X. Tang, "Learning a deep convolutional network for image super-resolution," in *Proc. European Conf. Computer Vision*, 2014.

[7] J. Kim, J. K. Lee, and K. M. Lee, "Accurate image super-resolution using very deep convolutional networks," in *Proc. IEEE Int'l Conf. Computer Vision and Pattern Recognition*, 2016.

[8] N. Yan, D. Liu, H. Li, and F. Wu, "A convolutional neural network approach for half-pel interpolation in video coding," in *Proc. IEEE Int'l Symposium on Circuits and Systems*, 2017.

[9] H. Zhang, L. Song, Z. Luo, and X. Yang, "Learning a convolutional neural network for fractional interpolation in HEVC inter coding," in *Proc. IEEE Visual Communication and Image Processing*, 2017.

[10] S. Xia, W. Yang, Y. Hu, and S. Ma, "A group variational transformation neural network for fractional interpolation of video coding," in *Proc. Data Compression Conference*, 2018.

[11] K. He, X. Zhang, S. Ren, and J. Sun, "Delving deep into rectifiers: Surpassing human-level performance on imagenet classification," in *Proc. IEEE Int'l Conf. Computer Vision*, 2015.

[12] D. Martin, C. Fowlkes, D. Tal, and J. Malik, "A database of human segmented natural images and its application to evaluating segmentation algorithms and measuring ecological statistics," in *Proc. IEEE Int'l Conf. Computer Vision*, 2001.

[13] Y. Jia, E. Shelhamer, J. Donahue, S. Karayev, J. Long, R. Girshick, S. Guadarrama, and T. Darrell, "Caffe: Convolutional architecture for fast feature embedding," in *Proc. ACM Int'l Conf. Multimedia*, 2014.

[14] D. Kingma and J. Ba, "Adam: A method for stochastic optimization," in *Proc. Int'l Conf. Learning Representations*, 2015.